Austin Neidert
August 21, 2018
MICPosPrinter Installation and Demonstration Notes.pdf 1.8

MICPosPrinter Introduction Tutorial

Two applications and a Visual Studio Project are required to follow this tutorial. The two applicaitons are the service object installer and the device manager application. The Visual Studio project contains a test program. The following document will act as an introduction to the Microcom Corporation POS for .NET Service Object.

Note: The MICPosPrinter service object has been developed for POS for .NET version 1.14. In order for the service object to function properly, POS for .NET 1.14 must be installed. It supports Microcom Printer Models 238, 338, and 438 with LDS2 support.

The service object acts as a full implementation of POS for .Net. This includes page mode, transaction mode, and in-line mode functionality along with the standard Open/Close, Claim/Release, Enable/Disable, CheckHealth, and DirectIO functionality. It also includes functionality for nearly every supplementary POS for .Net function.

In-line mode allows one print operation per line. These print operations will be buffered until 500 milliseconds pass between two print operations. After the 500 milliseconds pass, the buffered print operations are sent to the printer as a label.

Transaction mode only has two differences from in-line mode. The first difference is what causes the label to print. In transaction mode items are buffered until the TransactionPrint method is called. At that point (as opposed to after a 500 millisecond timeout) the label is printed. The second difference is the order of the items being printed. In in-line mode items are printed from the bottom up. This is because each print operation is treated as one “transaction”. In transaction mode the items are printed top to bottom, because the set of buffered print operations is one “transaction”.

Page mode is like transaction mode in the fact that print operations are buffered until the PageModePrint function is called. One difference page mode has from transaction mode is that items can be placed anywhere on the page in page mode (as opposed to printing down the left side of the page). Another difference is that in page mode the page size stays the same through print operations, whereas, in transaction mode and in-line mode page size starts at 0 and increases per print operation.

The Direct IO interface allows custom commands to be sent to the printer independent of the general POS printer functions. The DirectIO method provides a way to send arbitrary string data to the printer and receive the same from the printer as well as a way to send specific commands to the Service Object. The CheckHealth method issues a status request from the printer and should return “Ready” if the communication is working. CheckHealth can also be used to query the status of a specific feature; for example, the cutter status. This document provides a simple demo of some of the provided functionality.

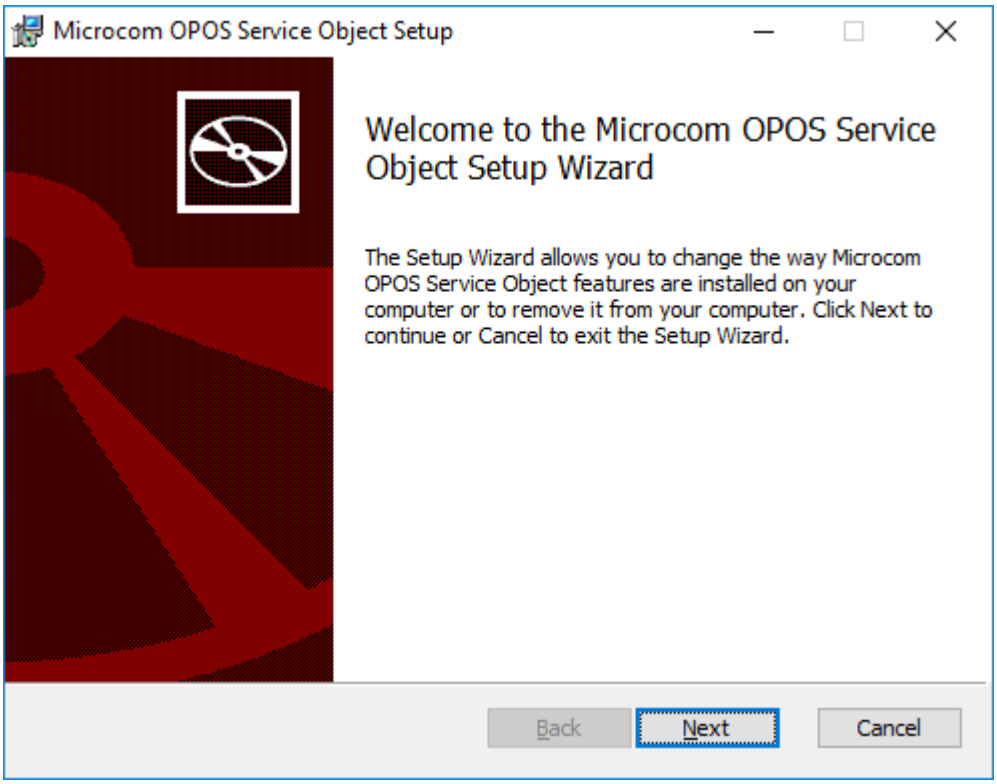
The device manager application acts as a GUI for modifying POS for .Net's “Configuration.xml” file. This can be used to add new logical devices or remove old logical devices. Once a logical device is added, this can be used to modify the device's general POS for .Net parameters such as the COM port. It can also be used to modify the device's parameters that are specific to the service object such as page width.

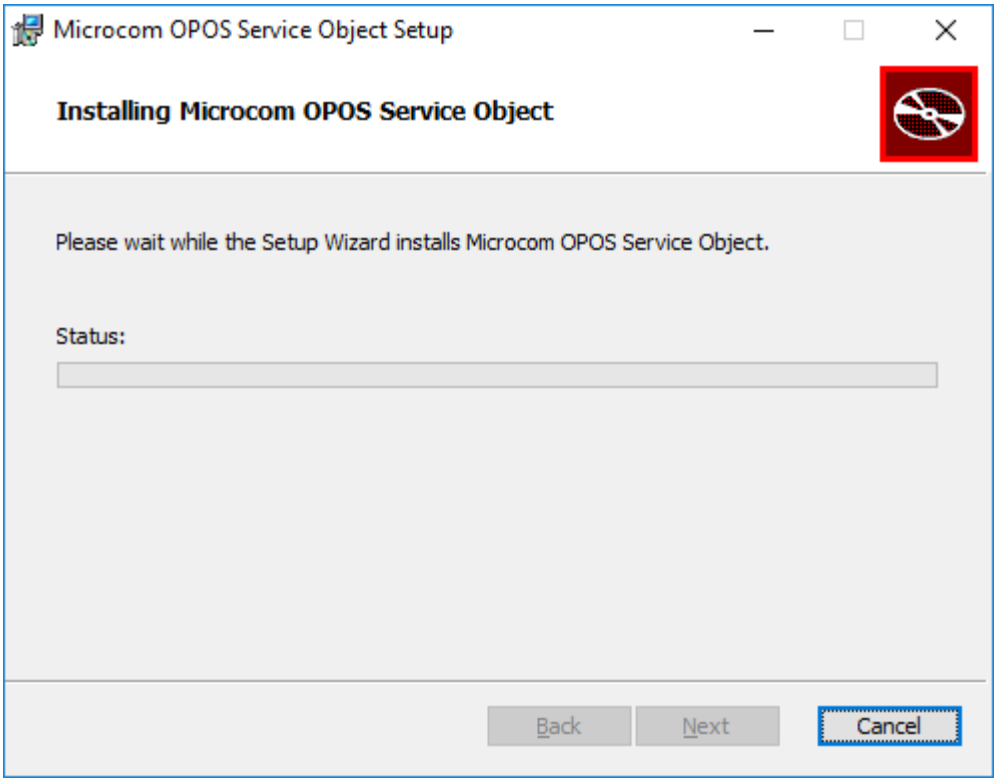
The Visual Studio project contains an application with a few demonstration functions. It provides easy access to basic commands. It contains buttons to demonstrate DirectIO, CheckHealth, in-line printing, transaction printing, and page printing. The in-line, transaction, and page printing code is referenced in the other included tutorials.

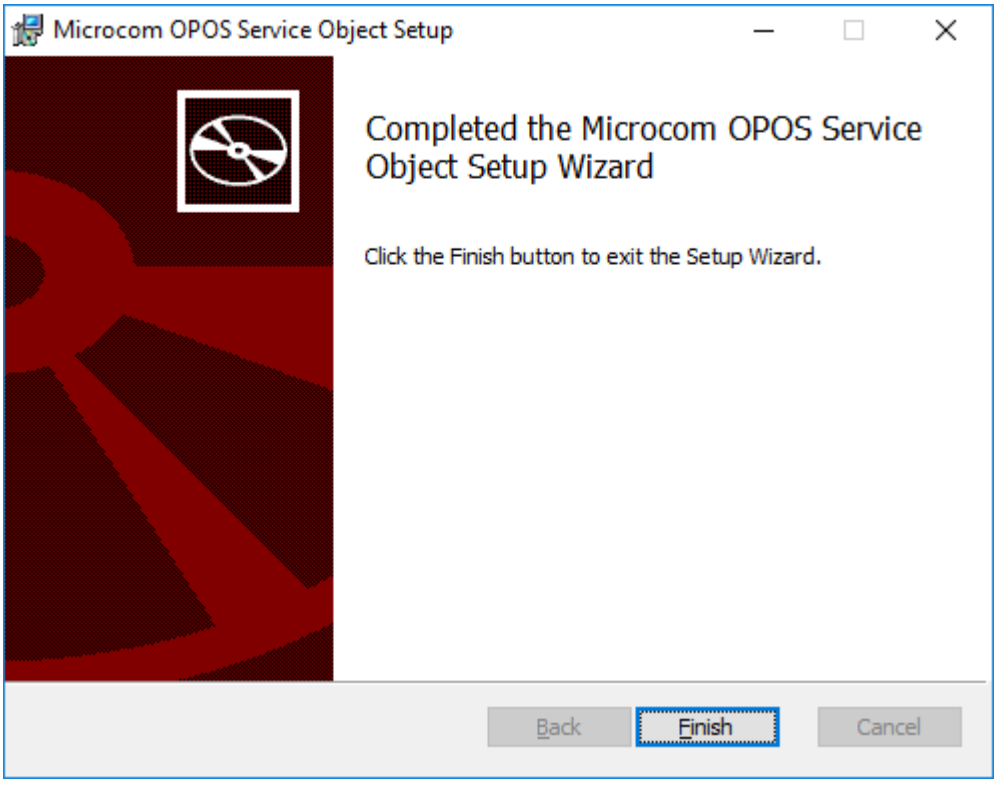
Note: All programs should be run as administrator.

Step 1: Installing the Service Object

Installing the service object can be done by running the installer. The installer is the file named “MicrocomPOSPrinterSO_Installer.msi”. Running the installer and following the prompts should yield the following screens.

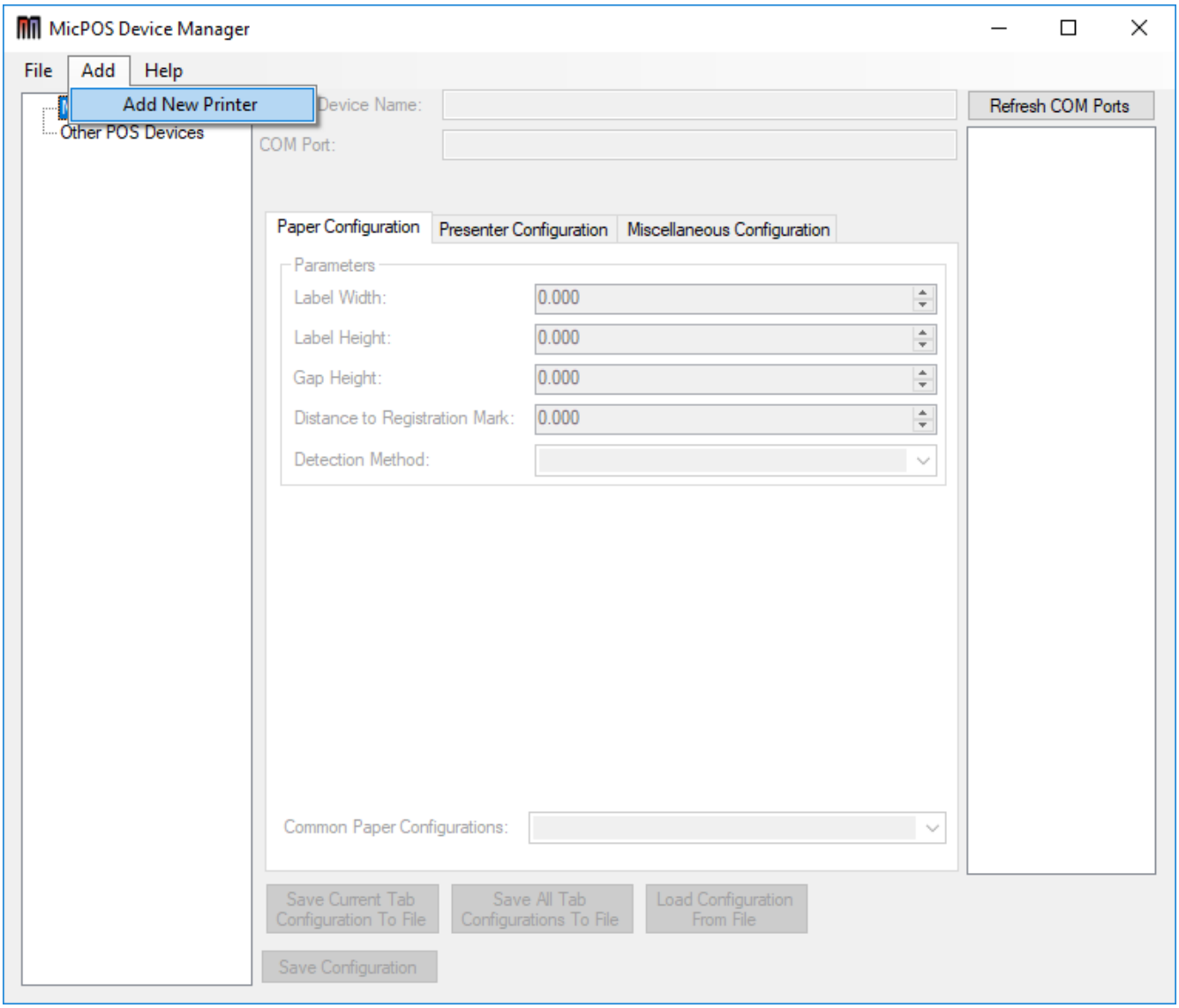



Microcom OPOS Service Object Setup
Installing Microcom OPOS Service Object
Please wait while the Setup Wizard installs Microcom OPOS Service Object.
Status:
Back
Next
Cancel

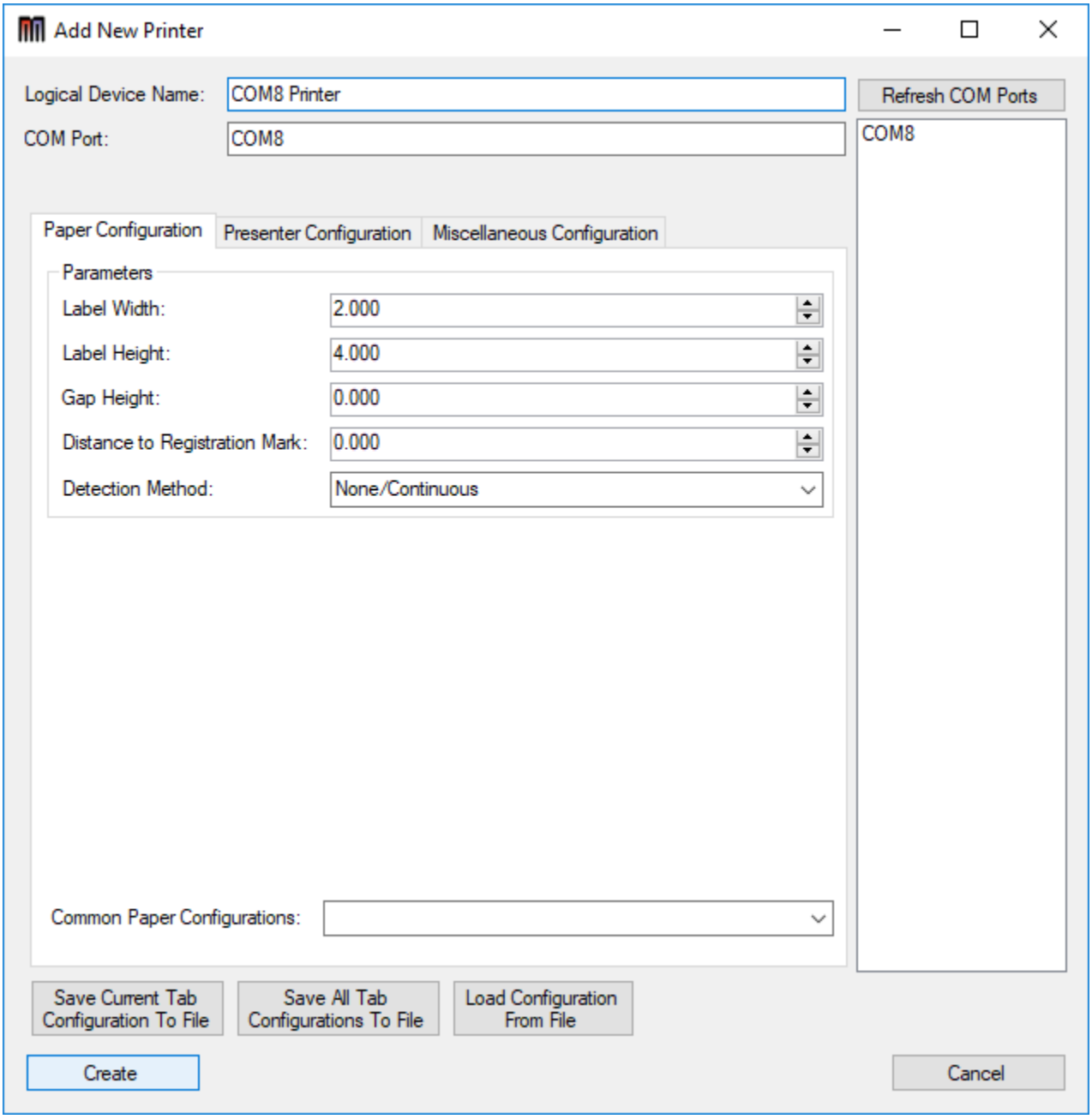

Microcom OPOS Service Object Setup
Completed the Microcom OPOS Service Object Setup Wizard
Click the Finish button to exit the Setup Wizard.
Back
Finish
Cancel

Step 2: Configure the Service Object

Configuring the service object can be done using the device manager application. First, run the “MicPOSDeviceManager.exe” file as administrator. Once the application is open, two entries should appear in the list on the left. Select the “Microcom POS Printers” entry. Once selected, click the “Add” button at the top followed by the “Add New Printer” button in the drop down menu.



After clicking the “Add New Printer” button, a new form should appear. In the “Add New Printer” form; enter the desired logical device name, enter the desired COM port, modify any other settings with undesired values, and click the “Create” button.



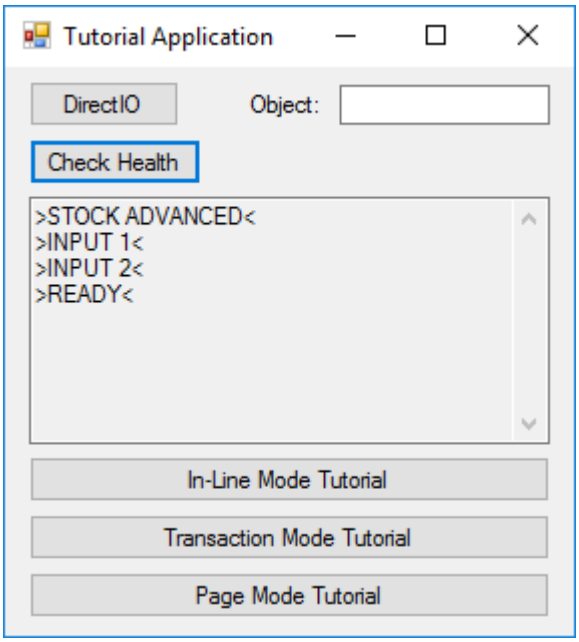
A dialog should appear confirming that the device was added successfully.

The service object is now installed and configured and the device manager application can now be closed.

Step 3: Test the Service Object

Step 3.1: CheckHealth

Open the Visual Studio project TutorialApplication.sln. Ensure that a printer is connected to the COM port entered in the device manager application and that the printer has power turned on. Then, run the application. On the generated form, click the “Check Health” button. The results should look like the following.

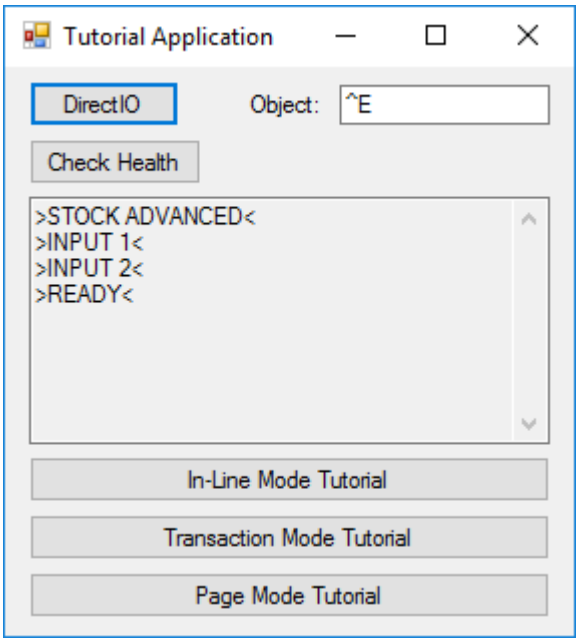


This demonstrates that there is communication between the service object and the printer.

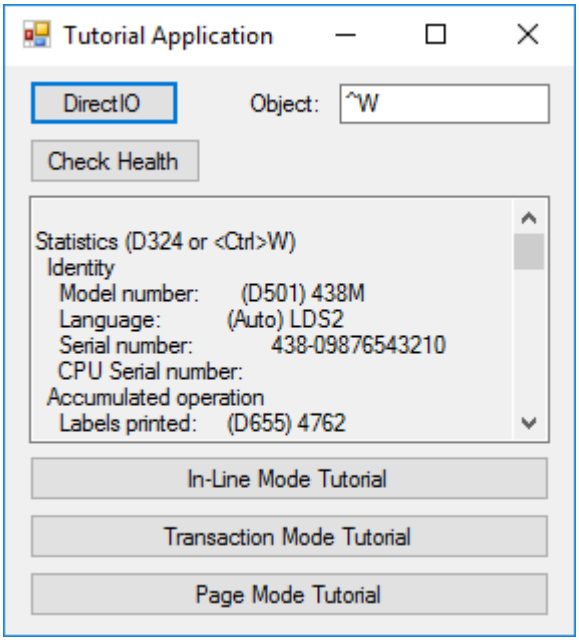
Note: The printer's status may be different than what is shown.

Step 3.2: DirectIO

In this tutorial application, modifying the command number can only be done through modification of code. Only command number zero (0) is important for now. This will send the Object field directly to the printer. In the Object field enter “^E” and then press the DirectIO button. The results should look like the following.



The “^E” command and the status response from the printer can be seen here. When the firmware is not being updated, calling CheckHealth sends “^E” to the printer. This means the DirectIO response will look identical to the CheckHealth response. Another command that could be tested is the “^W” command. This acts as a statistics command. The results should look like the following.



Note: The Object output field can be scrolled to examine the data returned, if it exceeds the size of the field.

This tutorial is complete and the Tutorial Application can now be closed.

The In-Line Mode, Transaction Mode, and Page Mode tutorials included are targeted toward POS for .NET developers as they reference .NET code. They explain how to use a Microcom POS for .NET printer programmatically. They reference the code called by the buttons in the tutorial application.

There is also a Test Application accompanied by its own reference document. This application is an extension of Microsoft's POS for .NET Test Application and can be used to test individual POS for .NET functions. It also has the capability of presenting a "page building" form to allow the user to use Page Mode without modifying code.